**JsonLabeller.py**

El script lee un archivo txt:

Este archivo puede contener una serie de numeros separados por comas.

La palabra all: Etiquetara a todos los ID’s como “Runners”

La palabra none: Etiquetara a todos los ID’s como “Public”.

Si el programa encuentra una serie de numeros, buscara dichos numeros (ID’s) en los archivos json del video correspondiente y cambiará sus etiquetas a “Runner”, además todo aquel ID que no se encuentre en el archivo TXT será modificado a “Public”.

Este script además se puede ejecutar de dos maneras:

-python JsonLabeller.py completefolder

-Esta opción etiquetara todos los videos encontrados en la carpeta JSON si poseen su respectivo TXT

-python JsonLabeller.py tgc-parquesur-clip13 tgc-parquesur-clip13

-Especificando primero la carpeta de los archivos JSON del video y segundo el archivo TXT del video

**JsonReaderV4.py**

El script lee los diferentes Json y nos da información sobre ellos según el comando utilizado:

-python JsonReaderV4.py VideoJsonsFolderName command

Commands:

info: Numero de archivos sin personas, Numero de archivos, Numero de personas detectadas y sus ID.

centroidfinder w h: Personas con un centroide cercano al introducido por parámetros.

idcentroidfinder id: Centroides del ID introducido.

Idfinder id: Datos de cada fotograma de ese ID.

type: Numero de corredores y de público, así como sus ID’s

**JsonAnalyser.py**

El script lee los archivos JSONs y genera un mapa de calor a partir del ultimo fotograma del video y de sus datos. Para poder utilizarlo debemos sacar ese último fotograma en formato png y guardarlo, con el mismo nombre que la carpeta JSON del video correspondiente, en la siguiente dirección: Deteccion-of-participants-in-sporting-events\yolov4-deepsort\outputs\Pictures\Originals.

-python JsonAnalyser.py commands:

completefolder centroidC: Genera los mapas de calor de todas las imágenes encontradas en la carpeta Originals mediante la impresión de círculos o puntos

completefolder centroidL: Genera los mapas de calor de todas las imágenes encontradas en la carpeta Originals mediante la impresión de líneas.

PictureName centroidC: Genera el mapa de calor del video especificado mediante la impresión de círculos o puntos.

PictureName centroidL: Genera el mapa de calor del video especificado mediante la impresión de líneas.

Todos los mapas de calor son guardados en la misma carpeta Pictures, y si se vuelven a generar estos se sobrescriben.

**JsonCharacteristic.py**

Este script lee todos los archivos JSONs, calcula las siguientes características: velocidad, velocidad promedio en 5 fotogramas, orientación y altura de la caja. Una vez obtenidas estas características son serializadas y guardadas en un archivo. Posteriormente discretizamos.

-python JsonCharacteristic process NumeroDeFotogramasPorSegundo: Generara dos archivos con los datos calculados, uno discretizados y otro normal.

-python JsonCharacteristic print standard: Se iniciará el script en modo visualización, permitiéndonos ver los datos calculados y sus cálculos de error.

En este modo se nos listara los videos analizados y podemos acceder a cada uno de ellos tecleando su numero o teclear “info” para ver el cálculo de errores.

Posteriormente podemos teclear “info” o un Id para ver su información.

Con el comando “back” podemos volver atrás y con el comando “q” detenemos la ejecución.

Para visualizar los datos discretizados cambiamos la palabra standard por discretise.

-python JsonCharacteristic print discretize

**TrainingData.py**

Este script utiliza los datos calculados y discretizados por el JsonCharacteristic.py y crea un panda dataframe. A su vez utiliza algoritmos clásicos de sklearn y entrena.

Lo primero es crear el dataframe, este proceso puede ser largo.

-python TrainingData.py process: Genera el archivo dataframe.pkl con el dataframe.

-python TrainingData.py train: Mediante el archivo dataframe.pkl entrena y muestra los resultados de múltiples algoritmos de sklearn.

- python TrainingData.py print: Muestra el dataframe.pkl.

**TrainingDataWithTrajectories.py**

Este script utiliza los datos calculados y discretizados por el JsonCharacteristic.py y crea un panda dataframe. A su vez utiliza algoritmos clásicos de sklearn y entrena.

El dataframe generado esta basado en trayectorias, contabilizando los datos de cada persona.

-python TrainingData.py process: Genera el archivo dataframe.pkl con el dataframe.

-python TrainingData.py train: Mediante el archivo dataframeWithTrajectories.pkl entrena y muestra los resultados de múltiples algoritmos de sklearn.

- python TrainingData.py print: Muestra el dataframeWithTrajectories.pkl.

**Lstm.py**

Este script crea el archivo dataframe.pkl si no existe y posteriormente entrena la lstm o carga una ya entrenada y muestra sus resultados.

-python lstm.py train FolderNameToSaveResults NumberOfEpochs: Entrena la LSTM con el número de épocas especificadas y guarda los resultados. Guarda en un archivo el estado de los pesos cuando se obtuvo el mejor valor de precisión en validación y en otro el estado final.

-python lstm.py load ‘. \models\FolderName\FolderNameMaxValAccuracy.h5’: Cargara los pesos del valor de precisión mayor.

-python lstm.py load ‘. \models\FolderName\FolderName.h5’: Cargara los pesos de la última época de entrenamiento.